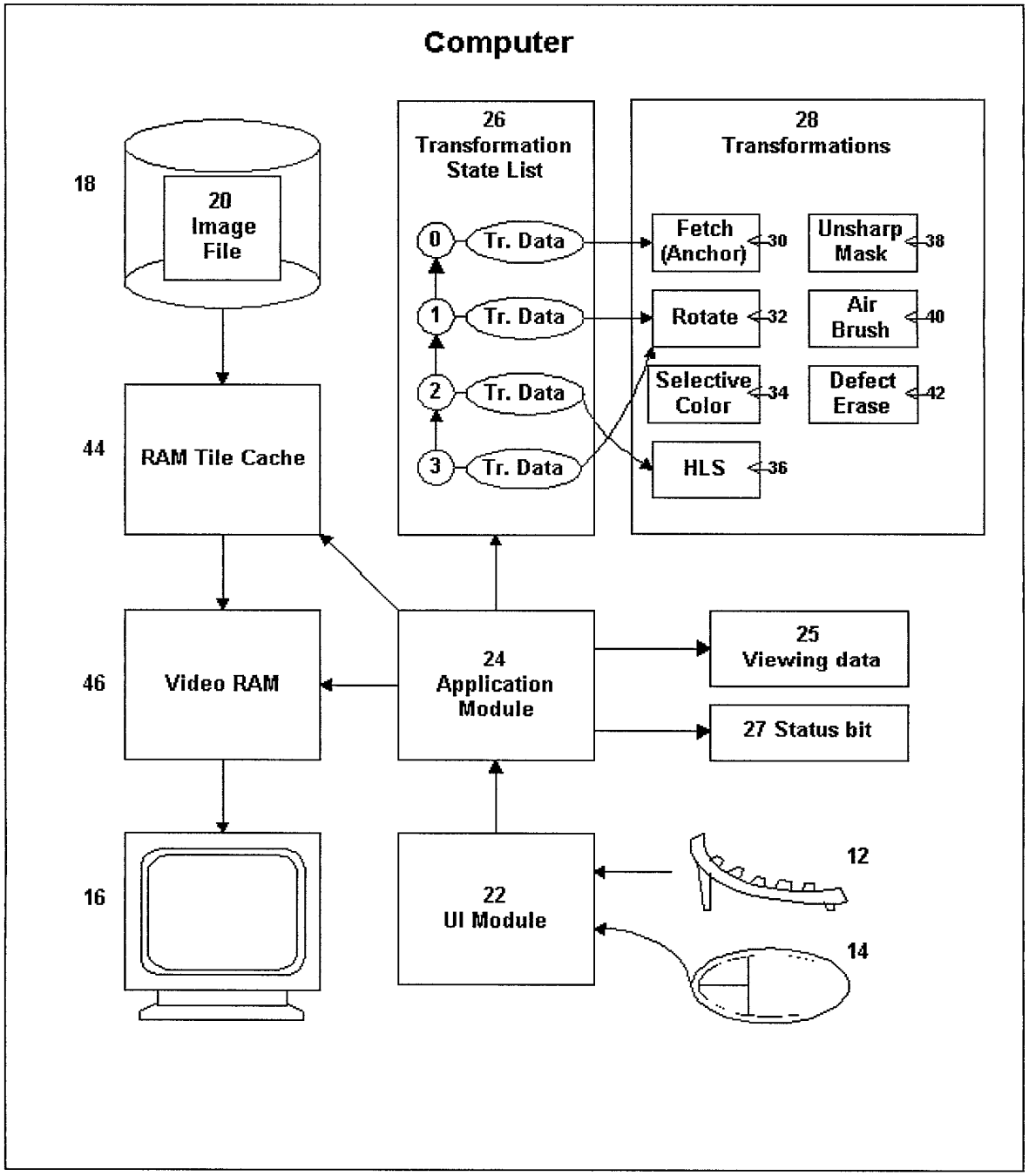


Figure 1
Fig1Standalone.vsd

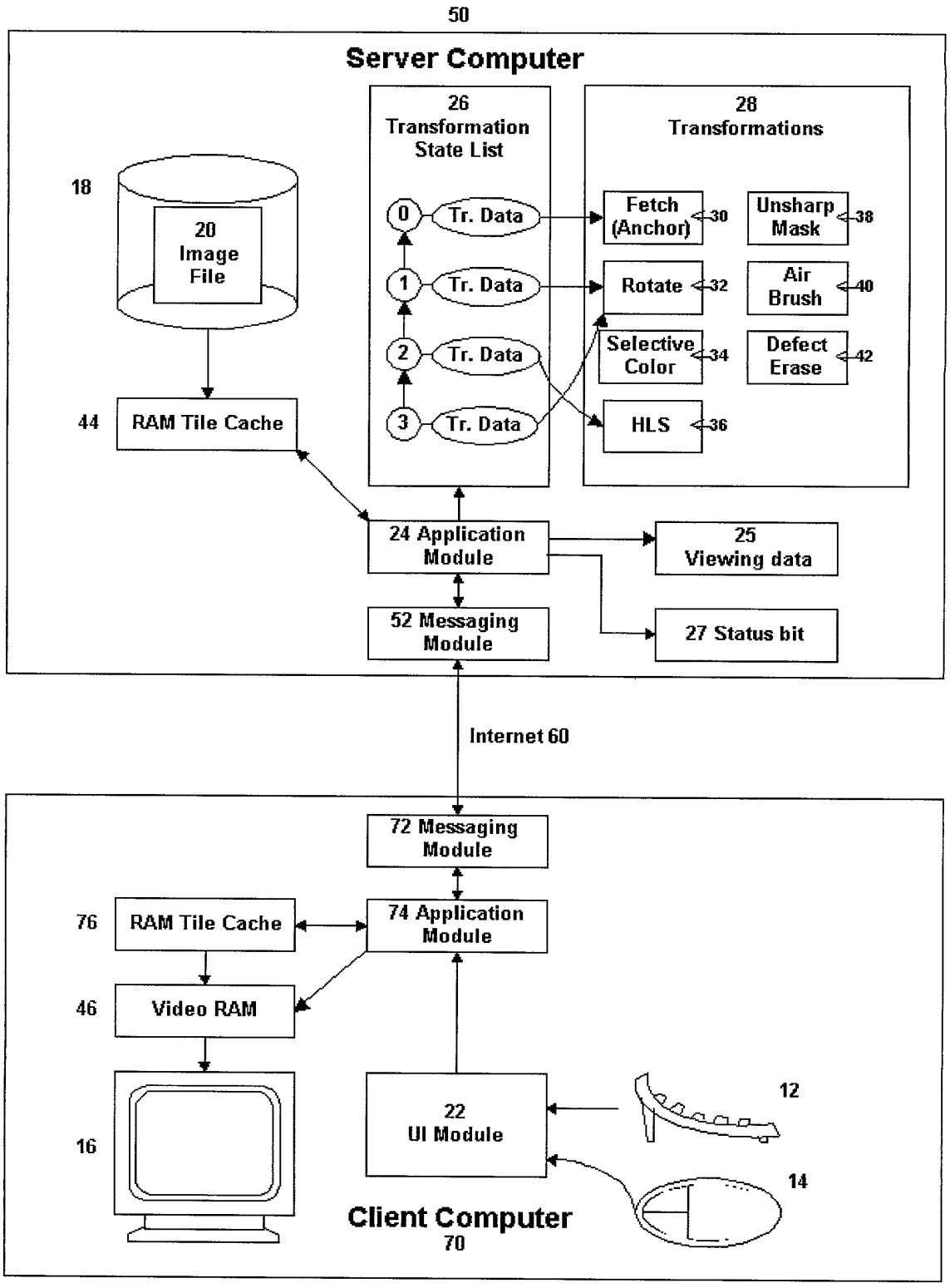
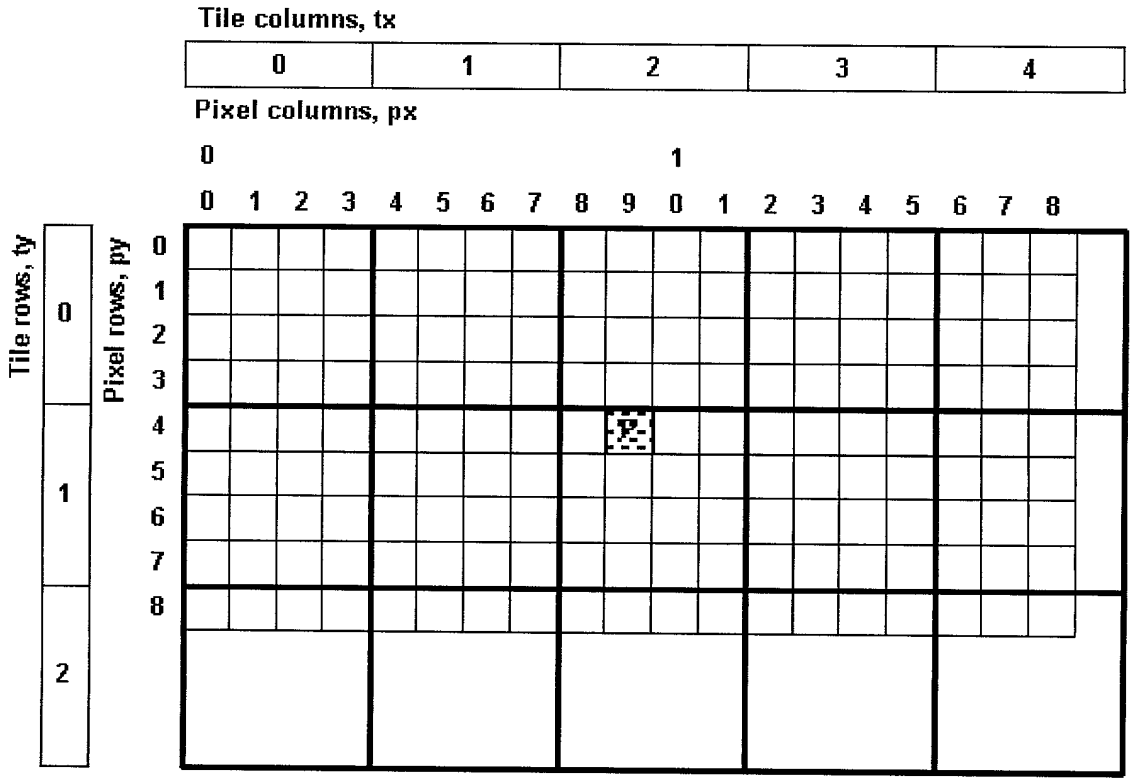


Figure 2

Fig2Network vsd



Pixel-Tile Conversion schema

(1) all tiles are of dimension n by n;
(2) the image origin is at the upper left;
(3) the coordinates of the pixel at the origin are (x=0, y=0);
(3) pixels are identified by their (column, row) pair, (x, y), offset from the origin;
(4) the pixel at the top left of the top-left tile is the origin pixel;
(5) tiles are identified by their (column, row) pair, (x, y), offset from the origin;
(6) n = 4;

Then:

the pixel P with coordinates (px, py) is in the tile T with coordinates
(tx, ty) = (px/n, py/n),
where the symbol "/" denotes integer division with the remainder discarded.

The coordinates (rx, ry) of pixel P relative to the origin of tile T are
(rx, ry) = (px%n, py%n),
where the symbol "%" denotes the remaider after integer division.

In the example shown, P has coordinates (px, py) = (9, 4);
hence it lies in tile T with coordinates (tx, ty) = (2, 1), and
its coordinates relative to the origin of that tile are (rx, ry) =(1, 0).

Figure 3

PixelTile.vsd

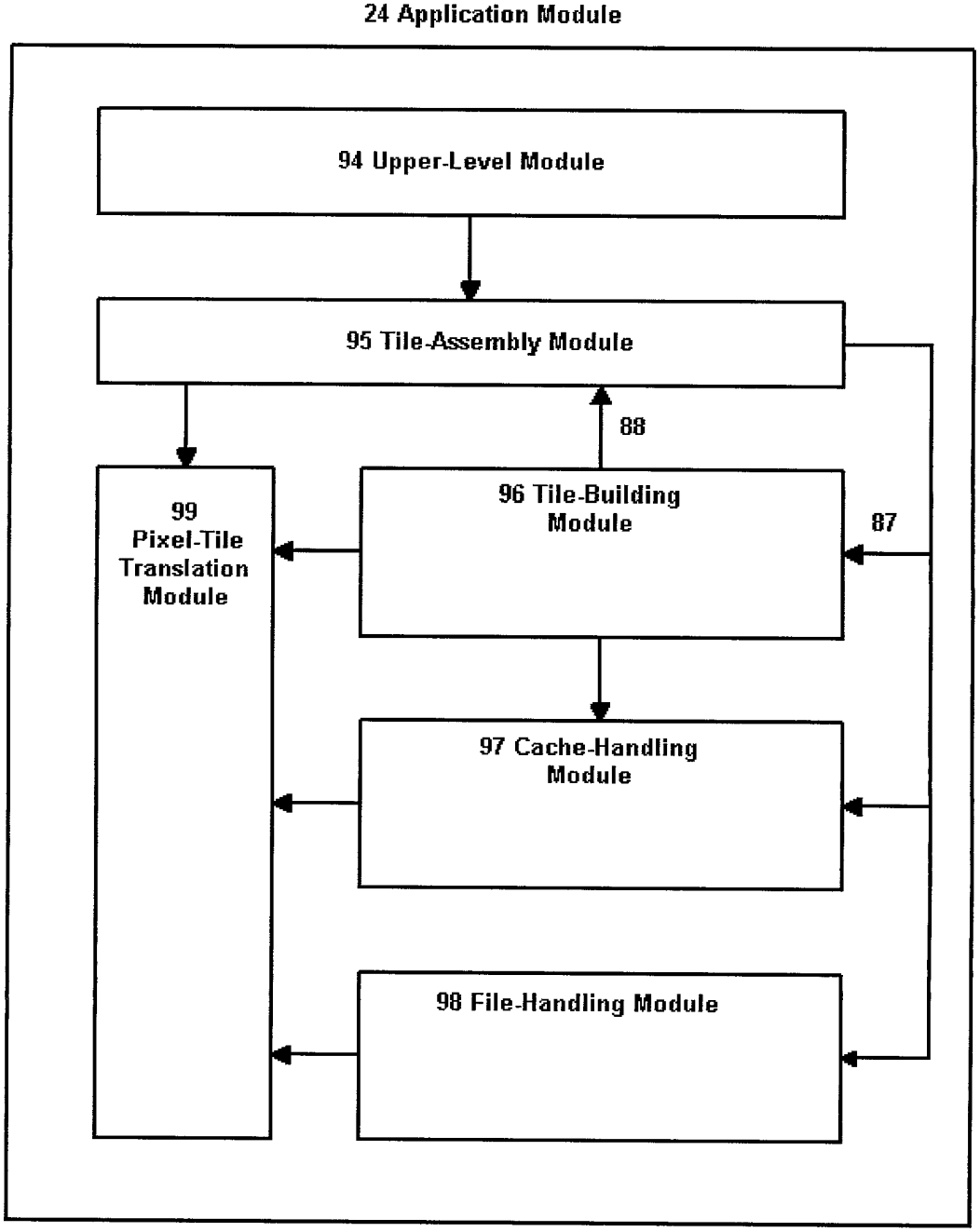


Figure 4
OverHier.vsd

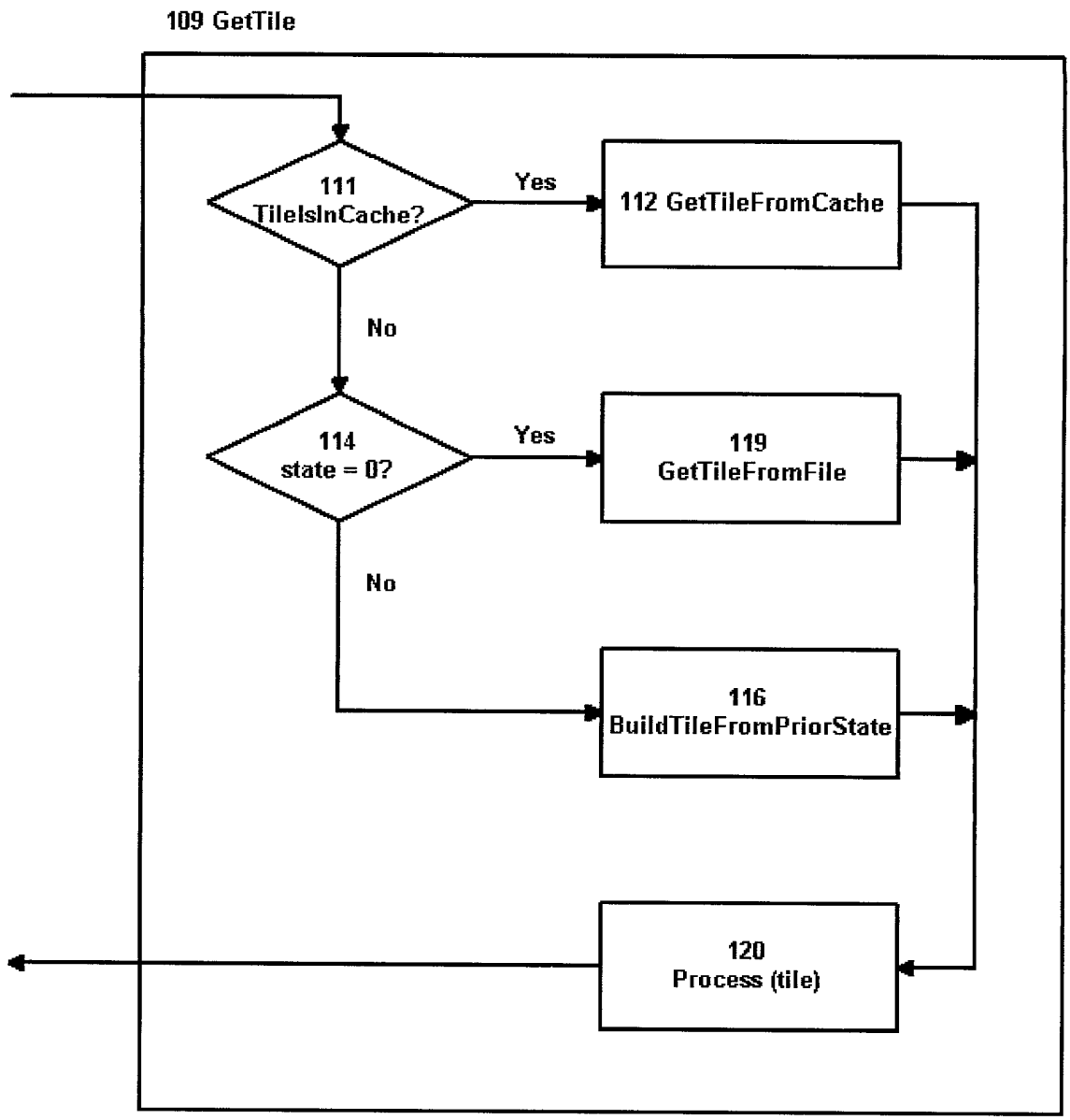


Figure 5
FlowBasic.vsd

```
100   // Figure 6a
101   GetTiles (state, scale, region)
102   {
103         ListOfTilesToGet = FindTilesIntersecting (state, scale, region);
104         For (all tiles in ListOfTilesToGet)
105               GetTile (state, scale, x, y);
106   }
107
108   // Figure 6b
109   GetTile (state, scale, x, y)
110   {
111         If (TileIsInCache (state, scale, x, y))
112               tile = GetTileFromCache (state, scale, x, y);
113         else
114         If (state != 0)
115               // comment: not the Anchor State
116               tile = BuildTileFromPriorState (state, scale, x, y);
117         else
118               // comment: Anchor State 0
119               tile = GetTileFromFile (scale, x, y);
120         Process (tile);
121   }
122
123   // Figure 6c
124   BuildTileFromPriorState (state, scale, x, y)
125   {
126         region = FindGeneratingRegion (state, scale, x, y);
127         GetTiles (state-1, scale, region);
128         tile = BuildTile (state, scale, x, y);
129         UnlockTiles (state-1, scale, region);
130   }
131
132   // Figure 6d
133   Process (tile)
134   {
135         If (!TileIsInCache (header (tile)))
136               CacheTile (tile);
137         LockTile (header (tile));
138   }
139
140
```

Figure 6, parts a-d

Fig6code.vsd

```
// Figure 6e
UnlockTiles (state, scale, region)
{
        ListOfTilesToGet = FindAllTilesIntersecting (state, scale, region);
        For (all tiles in ListOfTilesToGet)
                If (TileIsInCache (state, scale, x,  y))
                        UnlockTile (state, scale, x, y);
}

// Figure 6f
CacheTile (tile)
{
        If (!RoomInCache())
                PurgeTilesFromCache();
        AddTileToCache (tile);
}
```

Figure 6, parts e-f

Fig6Acode.vsd

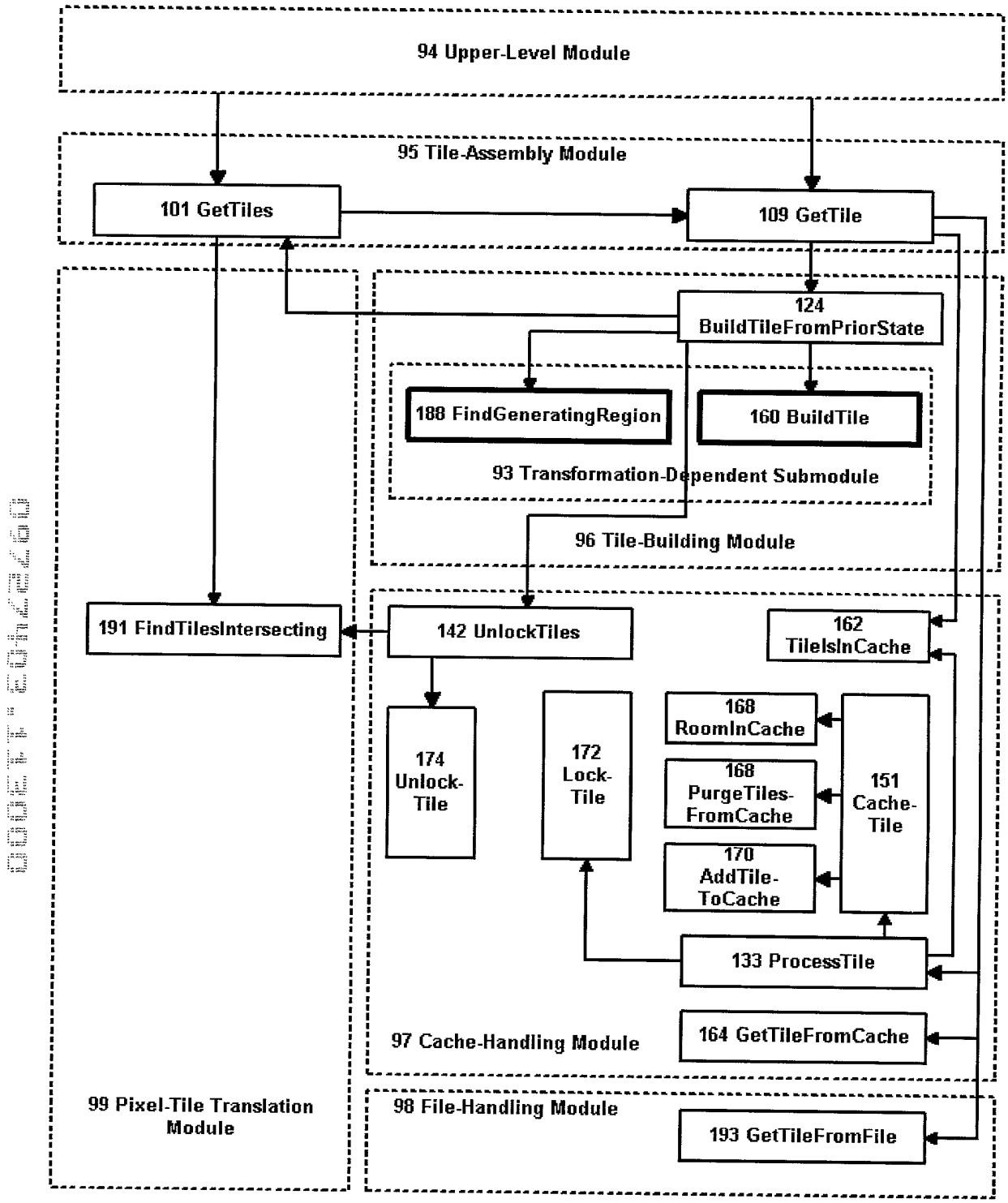
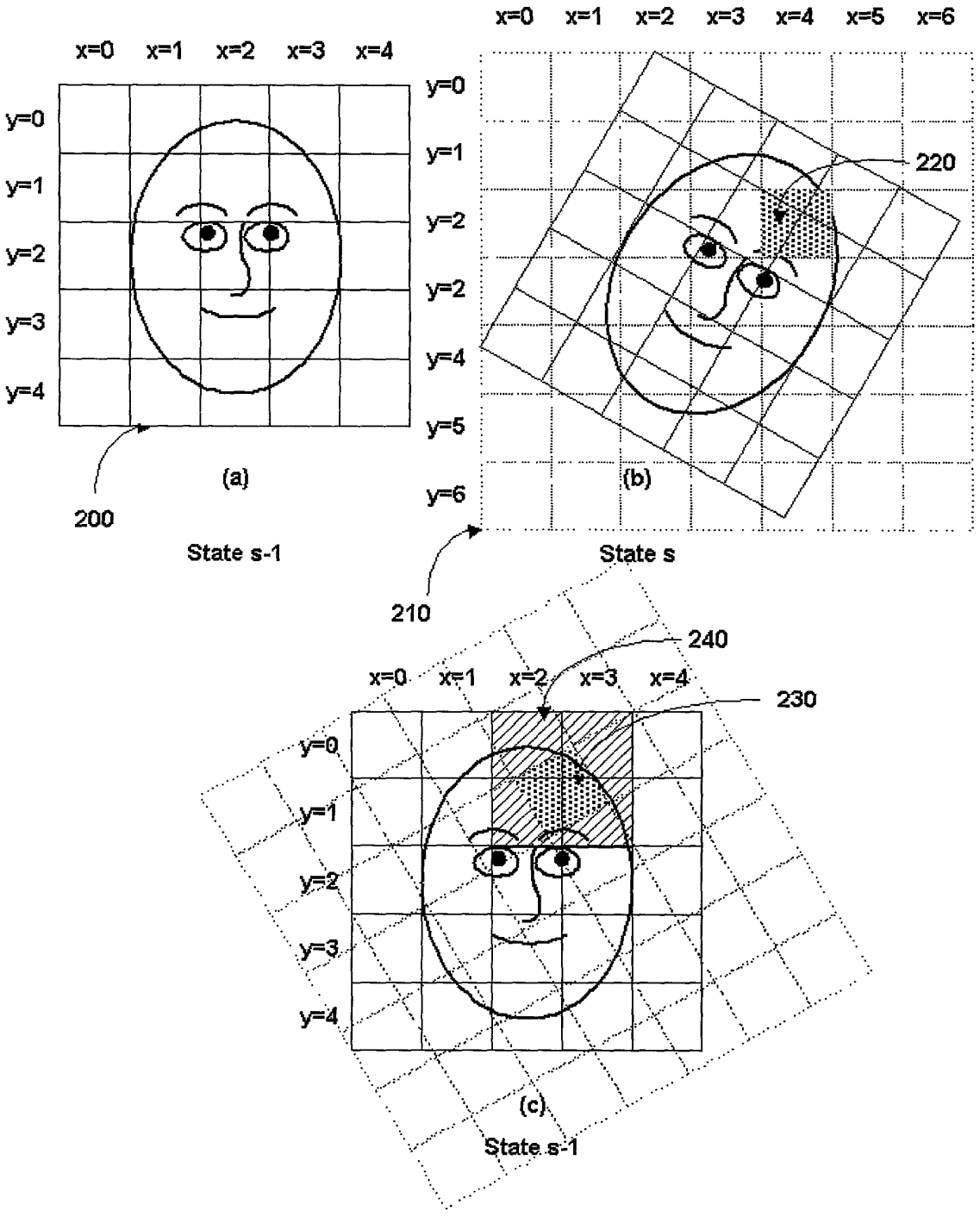


Figure 7
FuncHier.vsd



Inductive Image Generation

Figure 8

Rotate.vsd